(2)

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

AN ITERATIVE LINEAR PROGRAMMING APPROACH TO SOLVING LARGE CUMULATIVE SEARCH-EVASION GAMES

by

Brian P. Bothwell

March, 1990

Thesis Advisor: A. R. Washburn

Unclassified

security classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report Approved for public release; distribution is unlimited.
2b Declassification Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization Naval Postgraduate School 6b Office Symbol *(if applicable)* 30	7a Name of Monitoring Organization Naval Postgraduate School
6c Address *(city, state, and ZIP code)* Monterey, CA 93943-5000	7b Address *(city, state, and ZIP code)* Monterey, CA 93943-5000
8a Name of Funding Sponsoring Organization 8b Office Symbol *(if applicable)*	9 Procurement Instrument Identification Number
8c Address *(city, state, and ZIP code)*	10 Source of Funding Numbers Program Element No | Project No | Task No | Work Unit Accession No

11 Title *(include security classification)* AN ITERATIVE LINEAR PROGRAMMING APPROACH TO SOLVING LARGE CUMULATIVE SEARCH-EVASION GAMES

12 Personal Author(s) Brian P. Bothwell

13a Type of Report	13b Time Covered	14 Date of Report *(year, month, day)*	15 Page Count
Master's Thesis	From To	March 1990	48

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms *(continue on reverse if necessary and identify by block number)*
Field	Group	Subgroup	cumulative search-evasion games, game theory, linear programming

19 Abstract *(continue on reverse if necessary and identify by block number)*

Cumulative search-evasion games (CSEGs) involve two players, a searcher and an evader, who move among some finite set of cells. Neither player is aware of the other player's position during any stage of the game. When the payoff for the game is assumed to be the number of times the searcher and evader occupy the same cell, Eagle and Washburn proposed two solution techniques: one by fictitious play and the other by solving equivalent linear programming formulations. However, both have proved to be time consuming even for moderately sized problems.

This thesis considers two alternate linear programming formulations for CSEGs. Since both contain a large number of variables and constraints, the linear programming problems are initially solved with many of the constraints removed. If the solution to this relaxed problem is not a feasible optimal solution, additional constraints are added and the problem is solved again. This process continues until a feasible optimal solution is found. The results from a numerical experimentation with various solution techniques are also presented.

20 Distribution Availability of Abstract ☒ unclassified unlimited ☐ same as report ☐ DTIC users	21 Abstract Security Classification Unclassified	
22a Name of Responsible Individual A. R. Washburn	22b Telephone *(include Area code)* (408) 646-2786	22c Office Symbol 55Wa

DD FORM 1473,84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete. — security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

An Iterative Linear Programming Approach to Solving Large Cumulative Search-Evasion Games

by

Brian P. Bothwell
Lieutenant, United States Navy
B.S., University of Notre Dame, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 1990

Author: Brian P. Bothwell

Approved by: A. R. Washburn, Thesis Advisor

S. Lawphongpanich, Second Reader

Peter Purdue, Chairman,
Department of Operations Research

ABSTRACT

Cumulative search-evasion games (CSEGs) involve two players, a searcher and an evader, who move among some finite set of cells. Neither player is aware of the other player's position during any stage of the game. When the payoff for the game is assumed to be the number of times the searcher and evader occupy the same cell, Eagle and Washburn proposed two solution techniques: one by fictitious play and the other by solving equivalent linear programming formulations. However, both have proved to be time consuming even for moderately sized problems.

This thesis considers two alternate linear programming formulations for CSEGs. Since both contain a large number of variables and constraints, the linear programming problems are initially solved with many of the constraints removed. If the solution to this relaxed problem is not a feasible optimal solution, additional constraints are added and the problem is solved again. This process continues until a feasible optimal solution is found. The results from a numerical experimentation with various solution techniques are also presented.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

Cumulative search-evasion games (CSEGs) involve two players, a searcher and an evader, who move among some finite set of cells, C. Neither player is aware of the other player's position during any stage of the game. Let X_t and Y_t represent the positions of the searcher and evader, respectively, at time t. If the cells are numbered from 1 to n, then $X_t = i$ and $Y_t = j$ for some $i,j \in \{1,2,...,n\}$ for all t. The payoff of the game, N, is given by

$$N = \sum_{t=1}^{T} A(X_t, Y_t, t) \tag{1}$$

where T is the number of time periods and $A(X_t, Y_t, t)$ is the payoff function. The searcher wants to choose his strategy so as to maximize the expected payoff, E[N]; the evader desires to minimize the expected payoff. While there are many other suitable payoff functions, the one of interest is an indicator of the event $X_t = Y_t$. In this case, the payoff is equal to the average number of times that the two players occupy the same cell simultaneously.

This study assumes that the starting positions for both the searcher and the evader are specified. In particular, let S_0 and E_0 denote the starting locations (cells) of the searcher and evader, respectively. From one period to the next, both players are allowed to only move to cells which are adjacent to their currently occupied cells. In time period t, let the sets $S(X_t,t)$ and $E(Y_t,t)$ denote the cells which are adjacent to X_t and Y_t, then X_{t+1} and Y_{t+1} must belong to $S(X_t,t)$ and $E(Y_t,t)$, respectively. Figure 1 displays a one dimensional CSEG with four cells. If $X_t = 3$, then X_{t+1} must belong to $S(X_t,t) = \{2, 3, 4\}$. In essence, the players are not allowed to "leapfrog" to non-neighboring cells in one time period. Eagle and Washburn give several applications for CSEGs [Ref. 1].

Note that the objective function in a CSEG is not the probability of detection. In most search theory, the measure of effectiveness (MOE) to be maximized is the expected probability of detection. This MOE is most often given as $E[1 - e^{-N}]$, where N is as defined as above and $A(X_t, Y_t, t)$ is interpreted as the detection rate at time t. (See

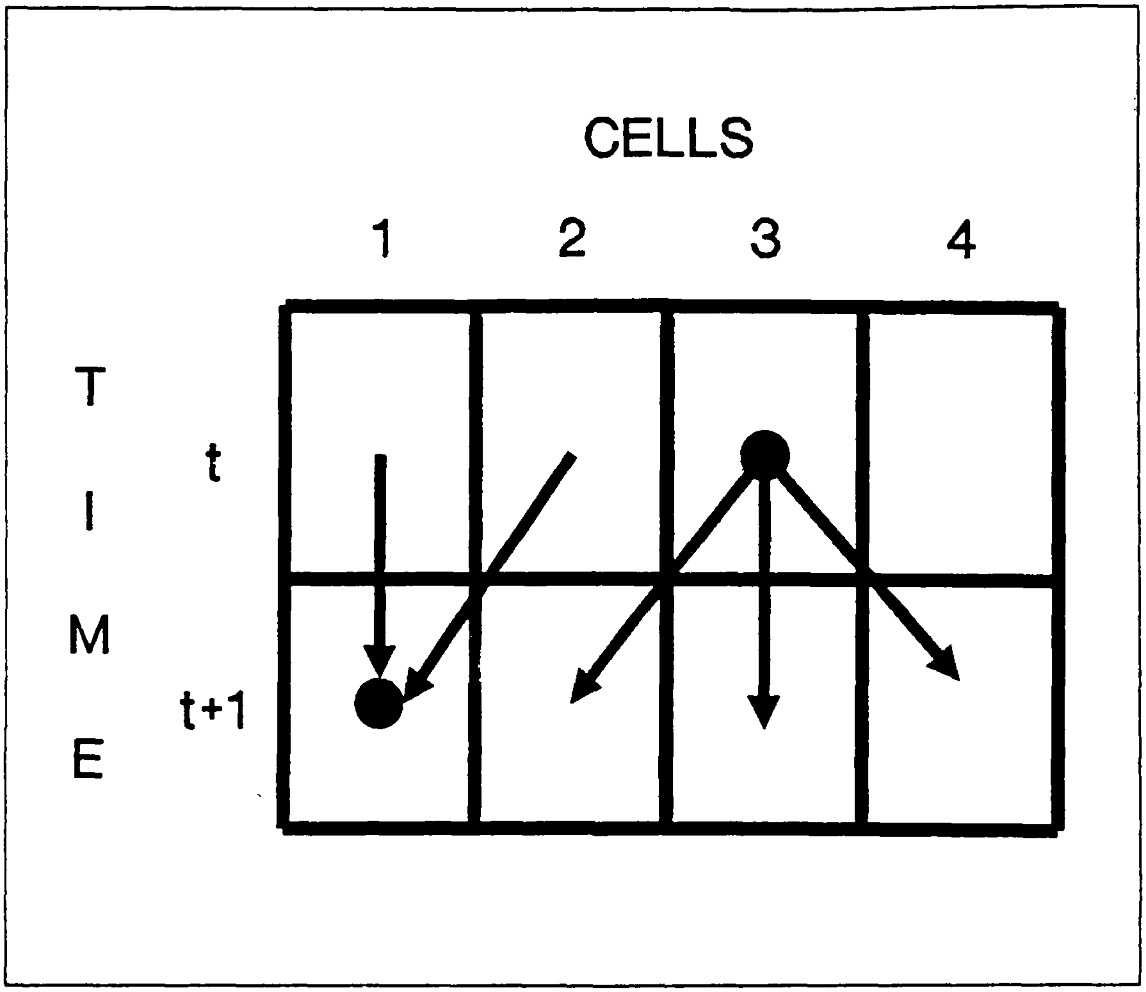

Figure 1. One Dimensional CSEG -- Motion Feasibility

Koopman[Ref. 2].) Since $E[1 - e^{-N}] \neq 1 - e^{-E[N]}$, the CSEG solution cannot be interpreted as a detection probability except when N is very small (Eagle and Washburn [Ref. 1]).

Most search theory deals with the optimal allocation of a searcher's effort to detect a target which is not allowed to actively evade. Most approaches (see Stone [Ref. 3]) rely on Bayesian methods for finding optimal search plans. Stone [Ref. 4] also surveys an extensive literature for finding optimal search allocation where the searcher is unconstrained and target motion follows a Markov process. Eagle [Ref. 5] and Stewart [Ref. 6] have studied the problem where the searcher is constrained.

Gal [Ref. 7] and Ruckle [Ref. 8] have considered games in which the target has been allowed to actively evade, using "time until capture" or "time until first detection" as the payoff. Stewart [Ref. 9] studied optimal search and evasion strategies for a two cell model under various constraints. Stewart's study assumes a detection rate payoff function which makes the analytical solution difficult for even the small model considered. CSEGs are thus an improvement on models such as Stewart's in the sense that games with several cells can be considered. The price paid for this is that CSEGs require a specific, analytically convenient form for the objective function.

II. EAGLE-WASHBURN FORMULATION

Eagle and Washburn [Ref. 1] offered two methods for solving CSEGs. One method is the Brown-Robinson method of fictitious play [Ref. 10]. The other approach is formulating the CSEG as a linear programming (LP) problem. Similar to the normal method of LP solution, there are two linear programming formulations: one for the searcher and the other for the evader. Both are equivalent in that they are duals of each other. However, unlike the normal method, where each pure strategy has a decision variable and a payoff matrix must be computed, this formulation is expressed in terms of marginal probabilities of the searcher occupying the cells in each time period t. The main motivation of this formulation is to avoid generating all possible pure strategies for both the searcher and evader. The number of pure strategies grows exponentially as a function of the length of the game, T, and the number of cells in $S(X_t, t)$ and $E(Y_t, t)$. For example, for a one dimensional search problem with 20 time periods and $S(X_t, t)$ and $E(Y_t, t)$ each containing three cells for each time period, there are 3^{20} pure strategies for each player.

To formulate the linear programming formulation for the searcher, Eagle and Washburn define the following:

$p(i,t)$ = the marginal probability that the searcher occupies cell i at time t

$u(i,j,t)$ = the probability that the searcher will visit cell i at time t and cell j at time t+1

$z(j,t)$ = the smallest possible payoff accumulated from period t to period T given that the evader occupies cell j in time period t

$S^{x}(i,t)$ = the set of cells in time period t-1 from which the searcher can reach cell i in time period t. That is,

$$S^{x}(i,t) = \{k \in C | i \in S(k,t-1)\}$$

for $i \in C$ and $t = 2,3,\ldots,T$, where $S(k,t-1)$ is as previously defined. For example, in Figure 1, $S^{x}(1,t+1) = \{1,2\}$.

Then, the linear programming problem can be stated as follows:

$$\max z(E_0,1)$$

subject to

$$p(S_0,1) = 1 \tag{2}$$

$$-\sum_{j\in S'(i,t)} u(j,i,t-1) + \sum_{k\in S(i,t)} u(i,k,t) = 0;\; i \in C,\; t = 2,3,\ldots,T-1 \tag{3}$$

$$-p(S_0,1) + \sum_{k\in S(S_0,1)} u(S_0,k,1) = 0 \tag{4}$$

$$-\sum_{j\in S'(i,T)} u(j,i,T-1) + p(i,T) = 0;\; i \in C \tag{5}$$

$$-\sum_{i\in C} A(i,j,t)p(i,t) - z(k,t+1) + z(j,t) \le 0;\; j \in C,\; k \in E(j,t),\; t = 1,2,\ldots,T-1 \tag{6}$$

$$-\sum_{k\in C} A(i,k,T)p(k,T) + z(i,T) = 0;\; i \in C \tag{7}$$

$$u(i,j,t) \ge 0;\; i,j \in C\; t = 0,1,\ldots,T-1 \tag{8}$$

The objective function above corresponds to maximizing the minimum payoff for the searcher given that the evader starts in cell E_0. Constraint (2) then restricts the searcher to start in cell S_0. To validate constraint (3), Eagle and Washburn observed that for $i \in C$ and $t = 1,2,\ldots,T-1$,

$$p(i,t) = \sum_{j\in S(i,t)} u(i,j,t) \tag{9}$$

or, alternatively

$$p(i,t) = \sum_{j \in S^*(i,t)} u(j,i,t-1). \tag{10}$$

Then, constraint (3) essentially enforces the equality of two equivalent expressions, (9) and (10), for p(i,t). There is also a network interpretation for constraint (3). Figure 2 depicts the network interpretation for a one dimensional CSEG with three cells and three time periods. The node (i,t) in the network represents cell i in the time period t and the flow on an arc connecting node (i,t) to (j,t+1) is represented by u(i,j,t). Then, constraint (3) is simply the conservation of flows at each node (i,t). Constraints (4) and (5) represent the terminal conditions for p(i,t) for t = 1 and t = T, respectively.

Based on the definition, z(j,t) can be written as

$$z(j,t) = \sum_{i \in C} A(i,j,t)p(i,t) + \min_{k \in L(j,t)} z(k,t+1) \tag{11}$$

where z(•,T+1) = 0. Constraint (6) is simply the linear representation of equation (11). Similar to constraints (4) and (5), constraint (7) is the terminal condition of equation (11).

The linear programming formulation for the evader is the dual of the above and is not presented here. The reader is referred to Eagle and Washburn [Ref. 1] for the details. However, it should be pointed out that the above formulation still contains a large number of decision variables, which in turn contributes to the extensive CPU time required to solve even a moderately sized CSEG. Most of these variables are u(i,j,t) variables. One objective of this thesis is to model the searcher's problem without these flow variables.

For the one dimensional CSEG, the formulation contains (3n-2)T flow variables and nT smallest payoff (z(i,t)) variables. (Marginal probability (p(i,t)) variables can be calculated from u(i,j,t) variables; they are not needed to solve the LP.) The number of constraints needed is approximately (4n-2)T + 2n.

The game reaches equilibrium when both searcher and evader marginal probabilities are uniform over cells 1,2,...,n. It can be shown that once the p(•,t) and q(•,t) reach this distribution, these distributions are optimal from that time onward (Eagle and

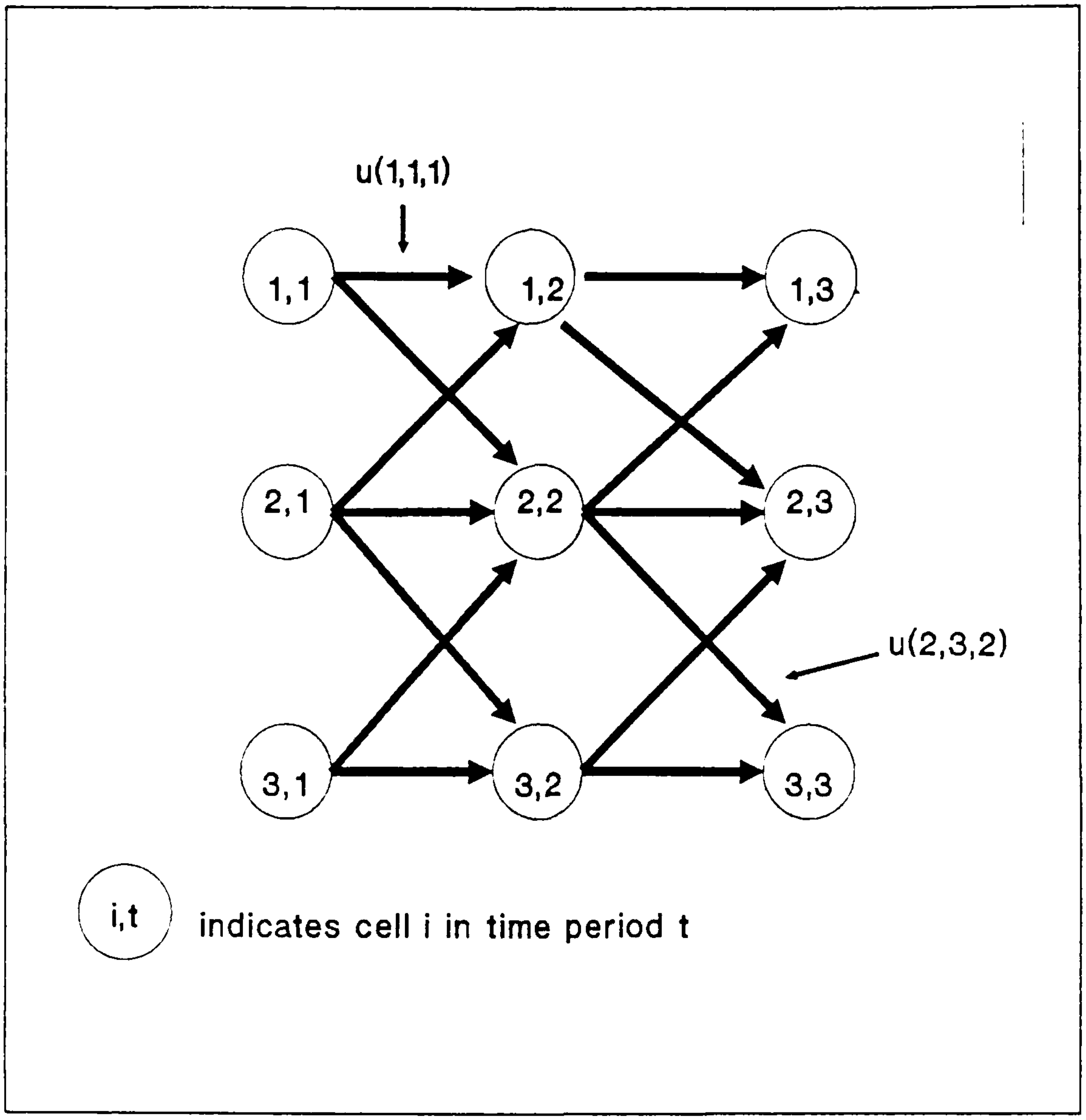

Figure 2. Network Flow Interpretation of a One Dimensional CSEG

Washburn [Ref. 1]). In order for the game to reach equilibrium, it was found by experiment that $T \approx 1.5n$. Thus, the Eagle-Washburn formulation requires on the order of $6n^2$ variables and constraints.

In the next chapter, two alternate linear programming formulations are considered. Both formulations contain fewer decision variables and many more constraints than the Eagle and Washburn formulation. However, the main advantage is the fact that many of the constraints are nonbinding. Thus, the problem size can be controlled by initially solving the linear program with only a small subset of the constraints and iteratively adding new constraints and resolving as necessary until an optimal solution is achieved.

III. METHODOLOGY

The Eagle and Washburn formulation always produces an exact solution for CSEGs. However, as the size of the problem grows, the time to solve the LP grows even more rapidly, making it impractical for solving large CSEGs. This chapter considers two alternate LP formulations based on preliminary work by Washburn and demonstrates how to solve these formulations in an iterative manner.

To simplify the presentation, only one dimensional CSEGs are considered. Also, it is assumed that

1. The search area consists of n cells.
2. The game is played for T time periods.
3. The searcher occupies cell 1 at time 1.
4. The evader occupies cell n at time 1.
5. The searcher and evader are each allowed to move one cell left or right or remain stationary at each time step of the game, i.e.,

$$S(i,t) = E(i,t) = \begin{cases} \{1,2\} & i = 1 \\ \{i-1, i, i+1\} & i = 2,3,\ldots,n-1 \\ \{n-1, n\} & i = n \end{cases}$$

A. METHOD ONE

Let q(j,t) be the marginal probability that the evader will be in cell j at time t. Then, the expected payoff, v, is given by

$$v = E[N] = \sum_{t=1}^{T} \sum_{i,j \in C} A(i,j,t) p(i,t) q(j,t) \tag{12}$$

where p(i,t) and N are as previously defined, A(i,j,t) = 1 iff i = j and E[N] denotes the expected value of N. The searcher wishes to maximize the expected payoff and the evader wishes to minimize the expected payoff.

Given that the marginal probabilities, p(i,t), are specified by the searcher, the evader then wants to find a strategy to minimize the payoff of the game. Let $Y = \{Y_1, Y_2, \ldots, Y_T\}$ be a feasible evader path or "track". If v represents the value of the game, then the following must hold for all strategies Y and marginal probability distributions p(i,t).

$$v \leq \sum_{t=1}^{T} \sum_{i \in C} A(i, Y_t, t) p(i,t) = \sum_{t=1}^{T} p(Y_t, t) \tag{13}$$

where the equality follows from the fact that $A(i,Y_t,t) = 1$ iff $i = Y_t$. Based on (13), the first alternate formulation can be written as

$$\max v$$

subject to

$$\sum_{i=1}^{n} p(i,t) = 1; \; t = 1,2,...,T \tag{14}$$

$$v \leq \sum_{t=1}^{T} p(Y_t, t) \; \forall \; Y \tag{15}$$

$$\sum_{i=k}^{l} p(i,t) \leq \sum_{j=k-1}^{l+1} p(j,t-1); \; k = 1,2,...,n, \; l = k,k+1,...,n, \; t = 2,3,...,T \tag{16}$$

$$p(i,t) \geq 0; \; i = 1,2,...,n, \; t = 1,2,...,T \tag{17}$$

Constraint (14) ensures that the marginal probability must sum to one for each time period and constraint (15) enforces the optimality condition expressed by equation (13). Finally, constraints (14) and (16) guarantee that there exists a corresponding set of feasible u(i,j,t). To illustrate that this is true, consider the CSEG with six cells and let k = 2 and l = 4. Then, constraint (16) translates to

$$p(2,t) + p(3,t) + p(4,t) \leq p(1,t-1) + p(2,t-1) + p(3,t-1) + p(4,t-1) + p(5,t-1)$$

which simply implies that the probability that the searcher will occupy cells 2, 3 or 4 at time t must not be larger than the probability that he will occupy cells 1, 2, 3, 4 or 5 at time t-1. If this condition does not hold, there can not exist a corresponding feasible "transition" probability, u(i,j,t). In the network interpretation, constraint (14) ensures that the total "supply" leaving nodes (i,t-1) and the total "demand" arriving at nodes (i,t) for i = 1,2,...,n is equal to one. Constraint (16) ensures that the amount of probability "shipped" along the arcs cannot exceed the "supply" of probability available.

Even for a moderately sized CSEG, the above formulation contains an extremely large number of constraints, in particular those which are described in constraints (15) and (16). Thus, it would be prohibitive to generate all the constraints and solve the resulting LP. Instead, the algorithm below initially solves the problem with one strategy Y = {n,n,...,n}, i.e., the evader remains in cell n for all T time periods, and disregards all type (16) constraints. Afterward, the violated constraints are added iteratively until all binding constraints are included in the formulation. The algorithm can be stated as follows:

Method One Algorithm

Step 0: Set k = 0 and let $(v^0, p^0(i,t))$ solve the following LP(0):

$$\max v$$

subject to

$$\sum_{i=1}^{n} p(i,t) = 1; \; t = 1,2,...,T \tag{18}$$

$$v \leq \sum_{t=1}^{T} p(n\ t) \tag{19}$$

$$p(i,t) \geq 0; \; i = 1,2,...,n, \; t = 1,2,...T \tag{20}$$

Step 1: If $(v^k, p^k(i,t))$ is feasible by constraint sets (15) and (16), then $(v^k, p^k(i,t))$ is a solution. Otherwise, go to Step 2.

Step 2: Generate constraints from the constraint sets (15) and (16) which $p^k(i,t)$ violates and add them to problem LP(k) to obtain a new problem LP(k + 1).

Step 3: Let $(v^{k+1}, p^{k+1}(i,t))$ solve LP(k + 1), set k = k + 1 and go to Step 1.

In Step 2, the feasibility of constraint set (16) is tested from level 1 to level n-2 for every time period until an infeasible condition is found or the solution can be declared feasible. The level refers to the number of cells considered. For example, the level 1 feasibility test consists of ensuring that p(i,t) ≤ p(i-1,t-1) + p(i,t-1) + p(i + 1,t-1) for i∈C and t = 2,3,...,T, where p(0,•) and p(n + 1,•) are defined to be zero. If this test is violated, a constraint of type (16) is added to the linear program for every such violation of level 1 feasibility tests and the LP is solved. If no violation is found, level 2 feasibility tests are performed. These tests ensure that p(i,t) + p(i + 1,t) ≤ p(i-1,t-1) + p(i,t-1) + p(i + 1,t-1)

$+\ p(i+2,t-1)$ for $i = 1,2,\ldots,n-1$ and $t = 2,3,\ldots,T$, where $p(0,\bullet)$ and $p(n+1,\bullet)$ are defined to be zero. These tests continue until a feasibility violation is found and the necessary constraints added or until the solution is found feasible. The number of these constraints could potentially grow as large as $\frac{1}{2}n(n+1)T$, but only a fraction of these constraints are needed.

Then, to test for feasibility for constraint (15), find a track, $\hat{Y}$, for the evader such that the sum of all $p(\hat{Y}_t,t)$ over all t is a minimum among all tracks. This can be accomplished by solving an appropriate shortest path problem. If

$$\sum_{t=1}^{T} p(\hat{Y}_t,t) \leq v^k \tag{21}$$

then constraint type (15) is violated and the constraint

$$v \leq \sum_{t=1}^{T} p(\hat{Y}_t,t) \tag{22}$$

must be added.

Method One was altered in two ways to attempt to reduce computation time. One variation involved eliminating slack constraints. Since each successive LP is larger than the previous LP, the time to solve each LP grows. By eliminating constraints that remain slack for several successive LP solutions, the overall size of the LP is reduced.

The other variation involved starting with a set of path constraints, or type (15) constraints, in addition to the usual starting constraints. If these path constraints can be chosen so as to cover the critical paths (path constraints that are tight in the final solution), this may reduce the number of algorithm iterations necessary to arrive at the problem solution.

B. METHOD TWO

The second method is similar to the first method, except that constraint set (15) is replaced by the recursive definition of the $z(\bullet,\bullet)$ variables stated in Chapter II. Using the fact that $A(i,j,t) = 1$ iff $i = j$, $z(\bullet,\bullet)$ can be redefined as follows:

$$z(i,t) = p(i,t) + \min_{k \in I(i,t)} z(k,t+1) \tag{23}$$

where $z(\cdot, T+1) = 0$. Both types of constraints serve the same purpose: to ensure that the game value is equal to the minimum payoff possible for a given $p(\bullet,\bullet)$. However, constraint set (15) was added as needed, while constraints corresponding to equation (23) are all in the initial LP. The advantage of the method is the fact that equation (23) produces (3n-2)T constraints while constraint set (15) contains 3^T.

Method Two Algorithm

Step 0: Let k = 0 and $(v^0, p^0(i,t))$ solve the following LP(0):

$$\max v$$

subject to

$$\sum_{i=1}^{n} p(i,t) = 1; \; t = 1, 2, \ldots, T \tag{24}$$

$$v = z(n,1) \tag{25}$$

$$z(i,t) \leq z(j,t+1) + p(i,t); \; i = 1,2,\ldots,n, \; j \in E(i,t), \; t = 1,2,\ldots,T-1 \tag{26}$$

$$p(i,t) \geq 0; \; i = 1,2,\ldots,n, \; t = 1,2,\ldots,T \tag{27}$$

Step 1: If $(v^k, p^k(i,t))$ is feasible to constraint set (16), $(v^k, p^k(i,t))$ is a solution. Otherwise, go to Step 2.

Step 2: Find constraints of type (16) which are violated and add them to problem LP(k) to obtain a new problem LP(k + 1).

Step 3: Let $(v^{k+1}, p^{k+1}(i,t))$ solve LP(k + 1), set k = k + 1 and go to Step 1.

Note that the method for finding violated constraints in Step 2 is as described for Method One.

Method Two was altered in two ways in an attempt to reduce computation time. The first variation was the method of eliminating slack constraints as done with Method One. The second variation made was adding a group of motion feasibility constraints, i.e., constraint set (16) to LP(0). This variation is similar to the second variation on Method One. If this group of motion feasibility constraints is chosen so as to cover those motion feasibility constraints which are tight in the solution, the number of algorithm iterations may be reduced. The method devised for choosing motion feasibility constraints for the starting group was drawn from examination of CSEG solutions obtained through the use of Method Two. This method proved very successful and is described below as Method Three.

C. METHOD THREE

Method Three is essentially the second variant of Method Two, except that the performance of Method Three is markedly better because it does not require successive iterations to arrive at an optimal solution. An examination of the tight motion feasibility constraints in optimal solutions using Method Two reveals that all these motion feasibility constraints are of the form:

$$\sum_{i=1}^{l} p(i,t) \leq \sum_{j=1}^{l+1} p(j,t-1); \; l = 1,2,\ldots,n-2 \tag{28}$$

All these constraints include cell 1, the left-most cell. These constraints will be referred to as left-anchored constraints.

Another observation from previous solutions is that the searcher always rushes from cell 1 to cell θ-1 during the first θ-1 time periods where $t = \theta$ is the first time period in which the searcher and evader can coincide. When there are n cells, $\theta = \left\lceil \frac{n}{2} \right\rceil$ when n is odd and $\theta = \frac{n}{2} + 1$ when n is even. Here [x] denotes the smallest integer m such that $x \leq m$. Since the searcher and evader cannot occupy the same cell during these first θ-1 time periods, $A(X_t, Y_t, t) = 0$ for $t = 1,2,\ldots,\theta-1$ for any pair of searcher and evader strategies (X,Y). Thus, no payoff occurs during periods 1 to θ-1 and z(n,1) must be the minimum of $z(i,\theta)$ where $i = \theta, \theta+1,\ldots,n$. This observation allows for ignoring all those decision variables from the first θ-1 time periods. This reduces the number of variables and constraints by approximately one-third.

Use of this formulation with left-anchored motion feasibility constraints will solve CSEGs as large as n = 12. In larger CSEGs, there exists another set of tight motion feasibility constraints which are anchored on the right-most cell, cell n. For these CSEGs, inclusion of this set of constraints for the last few time periods leads to an optimal solution. If we let τ = the first time period in which the right-anchored motion feasibility constraints are required, then the LP is as follows:

$$\max v$$

subject to

$$\sum_{i=1}^{n} p(i,t) = 1; \quad t = \theta, \theta+1, \ldots, T \tag{29}$$

$$z(i,t) \leq z(j,t+1) + p(i,t); \quad t = \theta, \theta+1, \ldots, T, \; i = 1,2,\ldots,n, \; j \in E(i,t) \tag{30}$$

$$v \leq z(i,\theta); \quad i = n+1-\theta, n+2-\theta, \ldots, n \tag{31}$$

$$\sum_{i=1}^{l} p(i,t) \leq \sum_{j=1}^{l+1} p(j,t-1); \quad t = \theta+1, \theta+2, \ldots, T, \; l = 1,2,\ldots,n-2 \tag{32}$$

$$\sum_{i=l}^{n} p(i,t) \leq \sum_{j=l-1}^{n} p(j,t-1); \quad t = \tau, \tau+1, \ldots, T, \; l = n, n-1, \ldots, 3 \tag{33}$$

$$p(i,t) \geq 0; \quad i = 1,2,\ldots,n, \; t = 1,2,\ldots,T \tag{34}$$

where $z(\bullet,T+1)=0$. Constraint (29) ensures the marginal probability summed over all cells in a time period is equal to one. Constraints (30) and (31) perform the same function as equation (23). Constraint (32) delineates all levels of left-anchored motion feasibility constraints for time periods of interest. Constraint (33) describes all right-anchored motion feasibility constraints for time periods of interest. The number of variables is $\frac{2}{3}nT+1$ and the number of constraints is on the order of $\frac{8}{3}nT$. Assuming $T \approx \frac{3}{2}n$, there are approximately n^2 variables and $4n^2$ constraints. This formulation provides solutions for CSEGs up to at least size $n=30$.

IV. RESULTS

Methods One and Two were both implemented through the use of a FORTRAN interface with LINDO (Linear Interactive Discrete Optimizer). Method Three and the Eagle-Washburn model were implemented in GAMS (General and Algebraic Model Solver) using MINOS (Modular In-Core Nonlinear Solver). All methods were executed on an IBM 3033AP mainframe computer at the Naval Postgraduate School, Monterey, California.

A. METHOD COMPARISON

Method One and its variants performed considerably worse than the Eagle-Washburn model for $n \geq 8$. Figure 3 shows computation time in CPU seconds versus CSEG size for both methods. Method Two is slightly faster than Method One, but does not perform as well as the Eagle-Washburn model for $n \geq 10$. In Figure 4, 2-1 indicates the performance of Method Two and 2-2 indicates the performance of the Method Two variant with slack constraint elimination.

Using Methods One and Two for solving larger CSEGs results in excessive computation time due to the increased number of LPs that must be solved. The strategy of solving several smaller LPs instead of one large LP fails because of the large number of small LPs that must be solved to arrive at the problem solution.

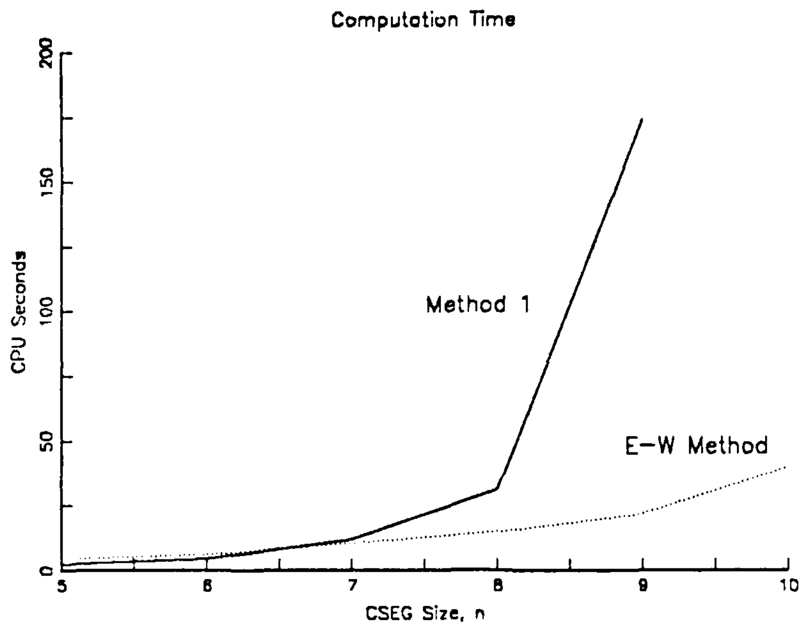

Figure 3. Computation Time of Method One vs. Eagle-Washburn Method

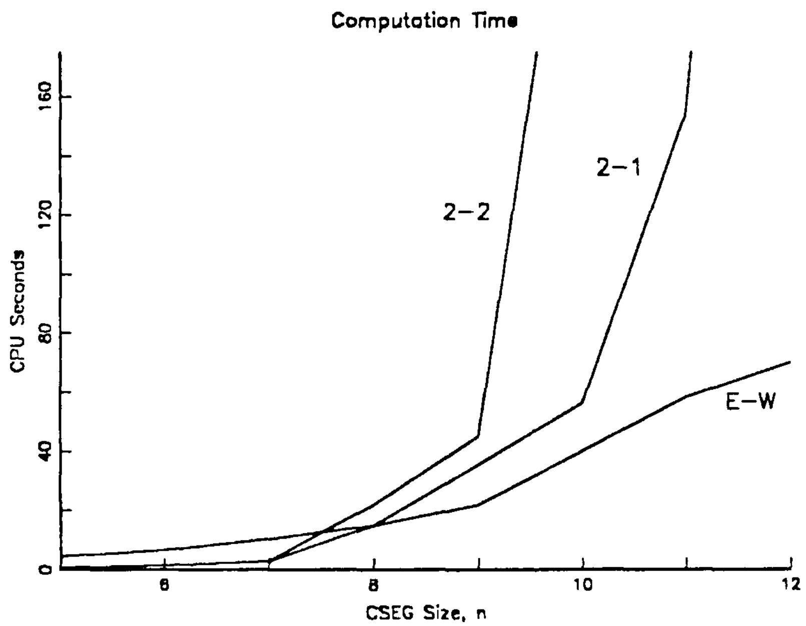

Figure 4. Computation Time of Method Two vs. Eagle-Washburn Method

Method Three proved to dominate the Eagle-Washburn model in all cases tested ($n \leq 30$). Figure 5 shows computation times for various size CSEG solutions under both models. Method Three allowed for solution of larger CSEGs than was previously economical using the Eagle-Washburn model. Solutions of these larger CSEGs have similar structure to smaller CSEG solutions, while showing some small differences.

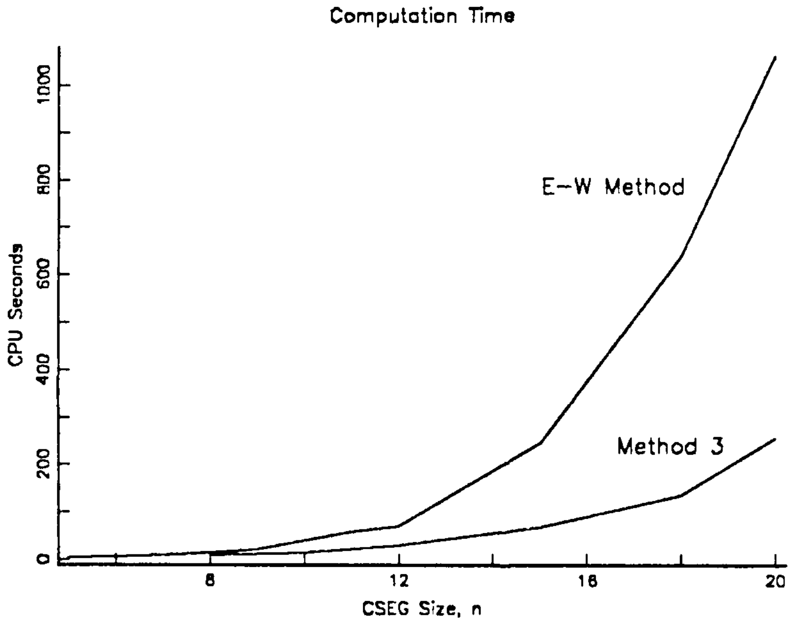

Figure 5. Computation Time of Method Three vs. Eagle-Washburn Method

B. CSEG SOLUTION STRUCTURE

Our concentration has been on finding the optimal searcher and evader marginal probabilities from the start of the game until the marginal probabilities reach equilibrium. The strategies of both players have been described previously by Eagle and Washburn [Ref. 1] for one-dimensional CSEGs as large as n = 12. Strategies in the larger games do not differ greatly from those in smaller games.

One pure strategy that might be advantageous to the evader includes staying in cell n until the searcher can reach that cell. If the game were played for less than n time periods, this would be the optimal strategy for the evader, since it would ensure a zero payoff. However, for longer games, this strategy is not optimal because of the large payoff the searcher can force when $t \geq n$. This strategy of waiting is part of a larger group of strategies that can be called "wait-and-run" strategies. The evader stays in cell n, or waits, for k time periods, where $k = 0,1,2,\ldots$. After he has waited k time periods, he moves to the left at top speed, one cell per time period, until reaching the left-most cells.

Figures 6 and 7 show that the evader mixed strategy consists of several "wait-and-run" pure strategies. If we interpret the probabilities as being parts of a large force, such as soldiers in an army, we can explain the results as follows. Note, in Figure 7, how 36 units break off immediately from the main force of 1000 units in the second time period. These 36 units continue to move left at the rate of one cell per time period until reaching cell 1 at time 20. This strategy corresponds to waiting zero time periods before running. Other "wait-and-run" strategies are used. In each successive period, the size of the force which breaks off from the main force in cell n increases. A large portion of the evader's force remains in cell n through the time period in which the searcher first arrives in cell n-1. At this point, the evader disperses this force from cell n as quickly as is feasible over the next few time periods. See Figure 8 for details of this strategy on an expanded scale.

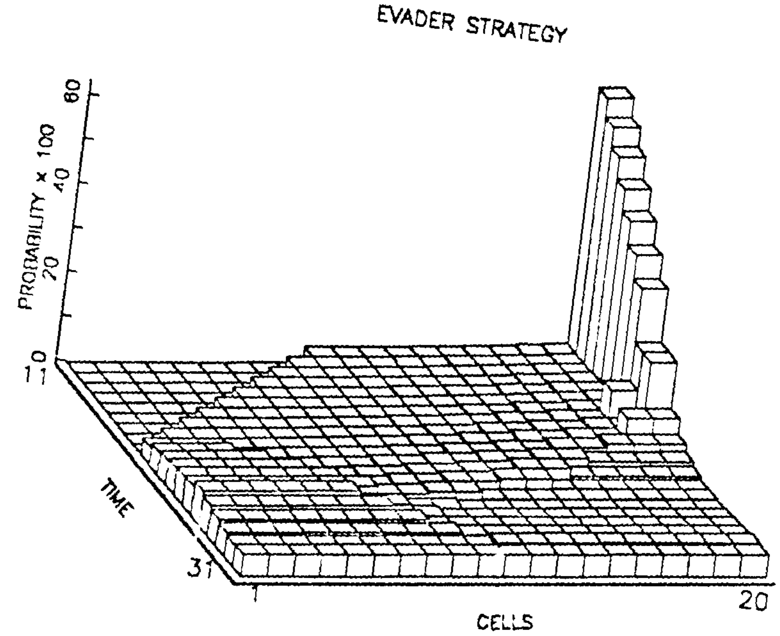

Figure 6. Evader Strategy

		CELLS																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1000
	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	964
	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	36	928
	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	36	36	892
	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	36	36	36	856
	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	36	36	36	37	819
	7	0	0	0	0	0	0	0	0	0	0	0	0	0	36	36	36	36	37	37	782
	8	0	0	0	0	0	0	0	0	0	0	0	0	36	36	36	36	37	37	39	743
	9	0	0	0	0	0	0	0	0	0	0	0	36	36	36	36	37	37	39	39	704
	10	0	0	0	0	0	0	0	0	0	0	36	36	36	36	37	37	39	39	41	663
	11	0	0	0	0	0	0	0	0	0	36	36	36	36	37	37	39	39	41	41	622
	12	0	0	0	0	0	0	0	0	36	36	36	36	37	37	39	39	41	41	44	578
	13	0	0	0	0	0	0	0	36	36	36	36	37	37	39	39	41	41	44	44	535
	14	0	0	0	0	0	0	36	36	36	36	37	37	39	39	41	41	44	44	49	485
T	15	0	0	0	0	0	36	36	36	36	37	37	39	39	41	41	44	44	49	49	436
I	16	0	0	0	0	36	36	36	36	37	37	39	39	41	41	44	44	49	49	54	382
M	17	0	0	0	36	36	36	36	37	37	39	39	41	41	44	44	49	49	54	54	329
E	18	0	0	36	36	36	36	37	37	39	39	41	41	44	44	49	49	54	54	121	208
	19	0	36	36	36	36	37	37	39	39	41	41	44	44	49	49	54	54	60	60	208
	20	36	36	36	36	37	37	39	39	41	41	44	44	49	49	54	54	60	60	104	104
	21	48	48	48	37	37	39	39	41	41	44	44	49	49	54	54	60	60	69	69	69
	22	54	54	54	54	39	39	41	41	44	44	49	49	54	54	60	60	52	52	52	52
	23	59	59	59	59	59	41	41	44	44	49	49	54	54	60	60	42	42	42	42	42
	24	63	63	63	63	63	63	44	44	49	49	54	54	60	60	35	35	35	35	35	35
	25	66	66	66	66	66	66	66	49	49	54	54	37	37	37	37	37	37	37	37	37
	26	78	78	78	78	78	78	49	49	39	39	36	36	36	36	36	36	36	36	36	36
	27	68	68	68	68	68	68	68	45	45	39	39	40	40	40	40	40	40	40	40	40
	28	59	59	59	59	59	59	59	59	45	45	44	44	44	44	44	44	44	44	44	44
	29	53	53	53	53	53	53	53	53	53	48	48	48	48	48	48	48	48	48	48	48
	30	47	47	47	47	47	47	47	47	47	47	53	53	53	53	53	53	53	53	53	53
	31	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50

Figure 7. Evader Marginal Probabilities (x1000) for 20-Cell CSEG.

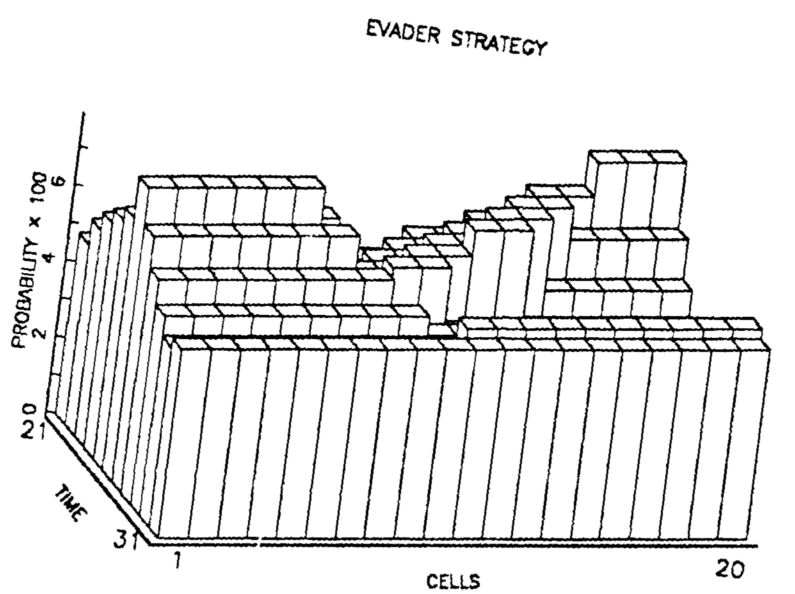

Figure 8. Evader Strategy - Final Time Periods

As the forces who ran approach cell 1, they join with the forces already there, i.e., those who ran before. Note in Figures 7 and 8 how the number of forces reaches a peak and then flattens out. Since both players can have forces in all cells at this point, the evader essentially disperses his forces to reach a uniform distribution over all cells.

Instead of using the individual soldiers interpretation, the explanation can also be made in terms of the probability of choosing a pure strategy from the optimal mixed strategy. The evader chooses from a number of "wait-and-run" strategies. It is more likely that he will wait for long periods of time before running than running early in the game. After reaching the left-most cells, it appears the evader's motion becomes more random as he spreads out towards a uniform distribution.

The searcher begins the game by rushing, or moving right at top speed, one cell per time period. His optimal strategy always consists of rushing over the first θ-1 periods where θ = the first time period in which the searcher and evader may occupy the same cell. Consider the five cell CSEG where the searcher begins in cell 1 and the evader begins in cell 5. Two time periods later, t=3, the searcher could travel as far right as cell 3, while the evader could travel as far left as cell 3. Thus, for the five cell CSEG, $\theta = 3$. This first possible meeting point is cell $\left\lceil \frac{n}{2} \right\rceil$ when n is odd and cell $\frac{n}{2}+1$ when n is even. The searcher gains nothing by stalling during these first time periods: for every time period he waits, he extends the number of zero payoffs he will receive.

During some time period (θ for CSEGs where n is even, $\theta+1$ for CSEGs where n is odd), the evader could for the first time be in a cell to the left of the searcher. For example, consider the six cell CSEG. At time 4, the searcher could be as far right as cell 4, the evader as far left as cell 3. If the players occupy those cells when t=4, then during time 3, the searcher and evader were in cells 3 and 4, respectively. Thus, at time 4, the searcher must split his forces between cells 3 and 4 to ensure that the evader cannot pass by without coincidence. This split of searcher forces arises in all CSEGs.

To return to using the analogy of individual soldier movement, after this initial split is made, the majority of the searcher's forces continue rushing towards cell n, while small forces split from this majority at every time step. See Figures 9 and 10. These small forces travel back towards cell 1, much as the evader's "wait-and-run" forces do. These small fractions of the searcher's forces make sure that the "wait-and-run" forces of the evader do not break through without paying some penalty. As with the evader's "wait-and-run" forces, the searcher's small split-off groups increase in size as the searcher nears cell n. Like the evader, once the searcher reaches cell n, he also disperses this main force as quickly as feasible. The forces then tend to move towards the uniform distribution. See Figure 11 for an expanded view of the searcher's strategy for the final time periods.

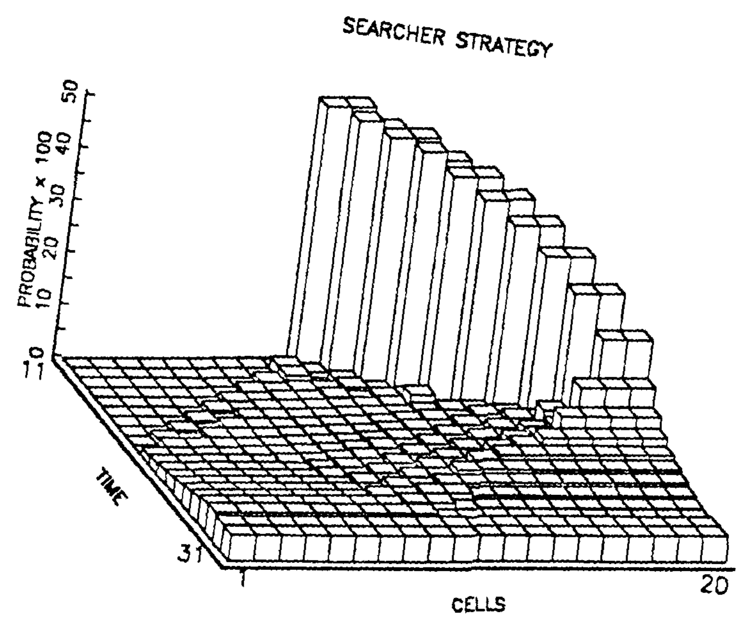

Figure 9. Searcher Strategy

CELLS

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1	1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	0	1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	3	0	0	1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	4	0	0	0	1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	5	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	7	0	0	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0	0	0	0
	8	0	0	0	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0	0	0
	9	0	0	0	0	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0	0
	10	0	0	0	0	0	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0
	11	0	0	0	0	0	0	0	0	0	500	500	0	0	0	0	0	0	0	0	0
	12	0	0	0	0	0	0	0	0	22	10	491	477	0	0	0	0	0	0	0	0
	13	0	0	0	0	0	0	0	10	22	0	13	477	477	0	0	0	0	0	0	0
	14	0	0	0	0	0	0	10	9	13	13	19	13	470	452	0	0	0	0	0	0
T	15	0	0	0	0	0	10	9	1	13	19	13	19	22	441	441	0	0	0	0	0
I	16	0	0	0	0	10	9	13	13	15	17	19	22	42	10	415	415	0	0	0	0
M	17	0	0	0	10	5	15	15	15	17	19	22	24	28	33	33	382	382	0	0	0
E	18	0	0	10	5	15	15	15	17	19	22	24	28	33	33	41	41	342	341	0	0
	19	0	10	5	15	15	15	17	19	22	24	28	33	33	41	41	52	52	290	290	0
	20	0	15	15	15	15	17	19	22	24	28	33	33	41	41	52	52	84	55	220	220
	21	15	15	15	15	17	19	22	24	28	33	33	41	41	52	52	66	73	147	147	147
	22	19	19	19	19	19	22	24	28	33	33	41	41	52	52	66	73	110	110	110	110
	23	24	24	24	24	24	24	28	33	33	41	41	52	52	66	73	88	88	88	88	88
	24	28	28	28	28	28	28	33	33	41	41	52	52	66	73	73	73	73	73	73	73
	25	34	34	34	34	34	34	34	41	41	52	52	64	64	64	64	64	64	64	64	64
	26	39	39	39	39	39	39	41	41	52	52	58	58	58	58	58	58	58	58	58	58
	27	45	45	45	45	45	45	45	52	52	58	58	52	52	52	52	52	52	52	52	52
	28	53	53	53	53	53	53	53	53	58	58	46	46	46	46	46	46	46	46	46	46
	29	56	56	56	56	56	56	56	56	56	45	45	45	45	45	45	45	45	45	45	45
	30	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50
	31	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50

Figure 10. Searcher Marginal Probabilities (x1000) for 20-Cell CSEG.

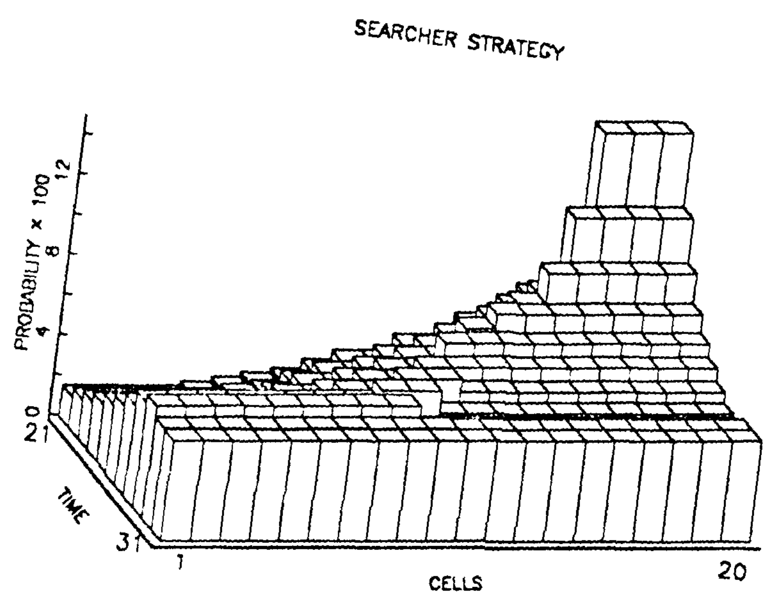

Figure 11. Searcher Strategy - Final Time Periods

Explained in terms of choosing a pure strategy from the optimal mixed strategy, the searcher's strategy always consists of rushing to the middle cells. As soon as he can possibly coincide with the evader, he has to make a choice of which direction to take.

Most often, he will continue to rush towards cell n. Sometimes, with a small probability, he will turn back and rush towards cell 1. At each time step, the searcher makes a decision on whether to reverse motion; each time the probability of reversing direction increases. Once the searcher has reached cell n (if he has chosen this strategy), his strategy is similar to the evader as both spread towards a uniform distribution over all cells.

Another striking point of the optimal solutions is the tendency of neighboring p(i,t) and q(i,t) values to be equal. These groups of equal value come most often in pairs; towards the end of the game, they come in larger groups. This is most easily seen in Figures 7 and 10, the tabular displays of searcher and evader probability distributions for the 20-cell CSEG. Although there are exceptions, these exceptions probably result from arriving at an alternate optimal solution. In working with small CSEGs that exhibit this exception, adding additional constraints to force equality among neighbors results in an alternate optimal solution.

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The formulation of Methods One and Two was intended to reduce computation time in two ways:

1. Reducing the number of decision variables by using marginal probability variables (p(i,t)) instead of probability flow variables (u(i,j,t)).
2. Reducing the work required to solve the game by reducing the number of constraints.

The switch to marginal probability variables reduces the number of variables needed to define the searcher's strategy by two-thirds for the one-dimensional game. A higher factor would apply in the two-dimensional game. However, the iterative method of selecting constraints for successive LP solutions proved to drastically increase the computation time necessary to solve larger games.

Inspection of the results obtained from various sizes of CSEGs solved with Methods One and Two led to the formulation of Method Three. With Method Three, computation time was further reduced by:

1. Eliminating decision variables for those time periods the searcher and evader cannot coincide.
2. Using those feasibility constraints of type (16) which contain p(1,t) for each time period.
3. Using those feasibility constraints of type (16) which contain p(n,t) for the final few time periods.

Eliminating the first few time periods in the one-dimensional CSEG reduces the number of variables by approximately one-third. The number of constraints is also reduced by one-third.

Method Three proves to be a faster solution method than the formulation of Eagle and Washburn. The use of just left- and right-anchored constraints for motion feasibility constraints is effective for $n \leq 30$.

B. RECOMMENDATIONS

The time required to solve CSEGs on the order of $n = 30$ still remains very large even with the reduction achieved with Method Three. Further reduction may be possible through the elimination of more variables or constraints.

In Methods Two and Three, all type (26) constraints are used. It may be possible to identify and eliminate those constraints which are always slack. Previously mentioned was the tendency of neighboring p(i,t) and q(i,t) values to be equal. The number of variables may be further reduced if the game can be modeled using pairs or groups of cells as decision variables. Extension of Model Three to the two-dimensional game should be attempted. The two dimensional CSEG would more closely model the real aspects of physical search than the one dimensional game. The solutions to CSEGs are very structured and there may be more ways of exploiting their characteristics to solve larger games more quickly.

APPENDIX GAMS PROGRAM OF METHOD THREE

```
$TITLE One-dimensional CSEG written by LT B.P. Bothwell      March 1990
$ONTEXT
This model uses the LP presented as Method Three.  It has been
proven to solve CSEGs up to a size of n=30.  The dimensions of the set I
(cells) is equal to n.  The dimension of the set T (time) is n+1 where
t=FIRST, FIRST+1,...,INT(1.5n)+1 where FIRST = the first time period in
which the searcher and evader may first coincide.  The model only
requires the motion feasibility constraints of level n-2 and lower.  For
example, if n=12, it is only necessary to include FEAS1 through FEAS10
and RFEAS1 through RFEAS10.  If it is desired to solve a CSEG of size
n>30, additional FEAS and RFEAS constraints must be added.  The program
displays searcher and evader marginal probabilities, game values, and
minimum and maximum possible payoff (z,ze) values.
$OFFTEXT
$OFFSYMXREF OFFSYMLIST OFFUELLIST OFFUELXREF
OPTIONS LIMROW=0,LIMCOL=0,SOLPRINT=OFF,RESLIM=3000,ITERLIM=12000
OPTION LP=MINOS5;
SETS
      I  cells    /C1*C12/
      T  time periods    /T7*T19/;
ALIAS (I,J);
PARAMETERS
      FIRST    first non-trivial time period
      ULTRA    first time period for right-handed constraints ;
$ONTEXT
      FIRST must be set to the first time period in which the searcher and
  evader can coincide.
      ULTRA is the first time period in which right-anchored constraints
  are used.  It is currently set to write these constraints for the last
  four time periods.
$OFFTEXT
```

```
FIRST=7;
ULTRA=CARD(T)+FIRST-4;
POSITIVE VARIABLES
      P(I,T)      searcher marginal distn in cell i at time t
      Z(I,T)      min value obtainable from t to T given evader in i at t;
$ONTEXT
      P(i,t) is fixed at zero if it is infeasible for the searcher
  to reach that cell
$OFFTEXT
P.FX(I,T)$(ORD(I) GT ORD(T)+FIRST-1) = 0;
VARIABLE
      V           game value;
$ONTEXT
      Equation description

      GAMEVAL constraints ensure v < z(i,t) for i > FIRST-1 and t = FIRST
      DIST* constraints ensure p(1,t)+p(2,t)+...+p(n,t)=1 for t>FIRST
      NET* constraints ensure z(i,t) < z(j,t+1) + p(i,t) for i in C,
  j in E(i,t) and t=FIRST,...,CARD(T)-1
      FEASa constraints ensure p(1,t)+...+p(a,t) < p(1,t-1)+...+p(a+1,t-1)
  for a=1,...,n-2 and t > FIRST+1
      RFEASa constraints ensure p(n,t)+...+p(n+1-a,t) < p(n,t-1)+...+p(n-a,t-1)
  for a=1,...,n-2 and t > FIRST+1
$OFFTEXT
EQUATIONS
      GAMEVAL(I,T)      game value constraints
      DISTONE(T)        inequality distn constraints
      DISTTWO(T)        equality distn constraint-final time period
      NETL(I,T)         intermediate network constraint type l
      NETE(I,T)         intermediate network constraint type e
      NETM(I,T)         intermediate network constraint type m
      FEAS1(T)          feasibility constraints for searcher marginals
      RFEAS1(T)
      FEAS2(T)
```

```
        RFEAS2(T)
        FEAS3(T)
        RFEAS3(T)
        FEAS4(T)
        RFEAS4(T)
        FEAS5(T)
        RFEAS5(T)
        FEAS6(T)
        RFEAS6(T)
        FEAS7(T)
        RFEAS7(T)
        FEAS8(T)
        RFEAS8(T)
        FEAS9(T)
        RFEAS9(T)
        FEAS10(T)
        RFEAS10(T)
        FEAS11(T)
        RFEAS11(T)
        FEAS12(T)
        RFEAS12(T)
        FEAS13(T)
        RFEAS13(T)
        FEAS14(T)
        RFEAS14(T)
        FEAS15(T)
        RFEAS15(T)
        FEAS16(T)
        RFEAS16(T)
        FEAS17(T)
        RFEAS17(T)
        FEAS18(T)
        RFEAS18(T);
$ONTEXT
```

```
     FEAS19(T)
     RFEAS19(T)
     FEAS20(T)
     RFEAS20(T)
     FEAS21(T)
     RFEAS21(T)
     FEAS22(T)
     RFEAS22(T)
     FEAS23(T)
     RFEAS23(T)
     FEAS24(T)
     RFEAS24(T)
     FEAS25(T)
     RFEAS25(T)
     FEAS26(T)
     RFEAS26(T)
     FEAS27(T)
     RFEAS27(T)
     FEAS28(T)
     RFEAS28(T) ;
$OFFTEXT
GAMEVAL(I,T)$((ORD(I) GE FIRST-1) AND (ORD(T) EQ 1))..
     V =L= Z(I,T) ;
DISTONE(T)$(ORD(T) LT CARD(T))..
     SUM(I,P(I,T)) =L= 100 ;
DISTTWO(T)$(ORD(T) EQ CARD(T))..
     SUM(I,P(I,T)) =E= 100 ;
NETL(I,T)$(ORD(I) GT 1)..
     Z(I,T) =L= Z(I-1,T+1) + P(I,T) ;
NETE(I,T)..
     Z(I,T) =L= Z(I,T+1) + P(I,T) ;
NETM(I,T)$(ORD(I) LT CARD(I))..
     Z(I,T) =L= Z(I+1,T+1) + P(I,T) ;
FEAS1(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
```

```
     SUM(I$(ORD(I) EQ 1),P(I,T)) =L= SUM(J$(ORD(J) LE 2), P(J,T-1)) ;
FEAS2(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$(ORD(I) LE 2),P(I,T)) =L= SUM(J$(ORD(J) LE 3), P(J,T-1)) ;
FEAS3(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$(ORD(I) LE 3),P(I,T)) =L= SUM(J$(ORD(J) LE 4), P(J,T-1)) ;
FEAS4(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$(ORD(I) LE 4),P(I,T)) =L= SUM(J$(ORD(J) LE 5), P(J,T-1)) ;
FEAS5(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$(ORD(I) LE 5),P(I,T)) =L= SUM(J$(ORD(J) LE 6), P(J,T-1)) ;
FEAS6(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 6)),P(I,T)) =L= SUM(J$((ORD(J) LE 7)),P(J,T-1)) ;
FEAS7(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 7)),P(I,T)) =L= SUM(J$((ORD(J) LE 8)),P(J,T-1)) ;
FEAS8(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 8)),P(I,T)) =L= SUM(J$((ORD(J) LE 9)),P(J,T-1)) ;
FEAS9(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 9)),P(I,T)) =L= SUM(J$((ORD(J) LE 10)),P(J,T-1));
FEAS10(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 10)),P(I,T)) =L= SUM(J$((ORD(J) LE 11)),P(J,T-1));
FEAS11(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 11)),P(I,T)) =L= SUM(J$((ORD(J) LE 12)),P(J,T-1));
FEAS12(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 12)),P(I,T)) =L= SUM(J$((ORD(J) LE 13)),P(J,T-1));
FEAS13(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 13)),P(I,T)) =L= SUM(J$((ORD(J) LE 14)),P(J,T-1));
FEAS14(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 14)),P(I,T)) =L= SUM(J$((ORD(J) LE 15)),P(J,T-1));
FEAS15(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 15)),P(I,T)) =L= SUM(J$((ORD(J) LE 16)),P(J,T-1));
FEAS16(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 16)),P(I,T)) =L= SUM(J$((ORD(J) LE 17)),P(J,T-1));
FEAS17(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 17)),P(I,T)) =L= SUM(J$((ORD(J) LE 18)),P(J,T-1));
FEAS18(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
```

```
     SUM(I$((ORD(I) LE 18)),P(I,T)) =L= SUM(J$((ORD(J) LE 19)),P(J,T-1));
$ONTEXT
FEAS19(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 19)),P(I,T)) =L= SUM(J$((ORD(J) LE 20)),P(J,T-1));
FEAS20(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 20)),P(I,T)) =L= SUM(J$((ORD(J) LE 21)),P(J,T-1));
FEAS21(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 21)),P(I,T)) =L= SUM(J$((ORD(J) LE 22)),P(J,T-1));
FEAS22(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 22)),P(I,T)) =L= SUM(J$((ORD(J) LE 23)),P(J,T-1));
FEAS23(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 23)),P(I,T)) =L= SUM(J$((ORD(J) LE 24)),P(J,T-1));
FEAS24(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 24)),P(I,T)) =L= SUM(J$((ORD(J) LE 25)),P(J,T-1));
FEAS25(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 25)),P(I,T)) =L= SUM(J$((ORD(J) LE 26)),P(J,T-1));
FEAS26(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 26)),P(I,T)) =L= SUM(J$((ORD(J) LE 27)),P(J,T-1));
FEAS27(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 27)),P(I,T)) =L= SUM(J$((ORD(J) LE 28)),P(J,T-1));
FEAS28(T)$((ORD(T) GE 2) AND (ORD(T) LT CARD(T)))..
     SUM(I$((ORD(I) LE 28)),P(I,T)) =L= SUM(J$((ORD(J) LE 29)),P(J,T-1));
$OFFTEXT
RFEAS1(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) EQ CARD(I)),P(I,T)) =L=
         SUM(J$(ORD(J) GE CARD(J)-1),P(J,T-1)) ;
RFEAS2(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-1),P(I,T)) =L=
         SUM(J$(ORD(J) GE CARD(J)-2),P(J,T-1)) ;
RFEAS3(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-2),P(I,T)) =L=
         SUM(J$(ORD(J) GE CARD(J)-3),P(J,T-1)) ;
RFEAS4(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-3),P(I,T)) =L=
```

```
        SUM(J$(ORD(J) GE CARD(J)-4),P(J,T-1)) ;
RFEAS5(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-4),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-5),P(J,T-1)) ;
RFEAS6(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-5),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-6),P(J,T-1)) ;
RFEAS7(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-6),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-7),P(J,T-1)) ;
RFEAS8(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-7),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-8),P(J,T-1)) ;
RFEAS9(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-8),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-9),P(J,T-1)) ;
RFEAS10(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-9),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-10),P(J,T-1)) ;
RFEAS11(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-10),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-11),P(J,T-1)) ;
RFEAS12(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-11),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-12),P(J,T-1)) ;
RFEAS13(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-12),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-13),P(J,T-1)) ;
RFEAS14(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-13),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-14),P(J,T-1)) ;
RFEAS15(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-14),P(I,T)) =L=
```

```
        SUM(J$(ORD(J) GE CARD(J)-15),P(J,T-1)) ;
RFEAS16(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-15),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-16),P(J,T-1)) ;
RFEAS17(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-16),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-17),P(J,T-1)) ;
RFEAS18(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-17),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-18),P(J,T-1)) ;
$ONTEXT
RFEAS19(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-18),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-19),P(J,T-1)) ;
RFEAS20(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-19),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-20),P(J,T-1)) ;
RFEAS21(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-20),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-21),P(J,T-1)) ;
RFEAS22(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-21),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-22),P(J,T-1)) ;
RFEAS23(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-22),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-23),P(J,T-1)) ;
RFEAS24(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-23),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-24),P(J,T-1)) ;
RFEAS25(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-24),P(I,T)) =L=
        SUM(J$(ORD(J) GE CARD(J)-25),P(J,T-1)) ;
RFEAS26(T)$(ORD(T) GE ULTRA)..
     SUM(I$(ORD(I) GE CARD(I)-25),P(I,T)) =L=
```

```
        SUM(J$(ORD(J) GE CARD(J)-26),P(J,T-1)) ;
RFEAS27(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-26),P(I,T)) =L=
         SUM(J$(ORD(J) GE CARD(J)-27),P(J,T-1)) ;
RFEAS28(T)$(ORD(T) GE ULTRA)..
      SUM(I$(ORD(I) GE CARD(I)-27),P(I,T)) =L=
         SUM(J$(ORD(J) GE CARD(J)-28),P(J,T-1)) ;
$OFFTEXT
MODEL CSEG /ALL/ ;
SOLVE CSEG USING LP MAXIMIZING V ;
$ONTEXT
  DISPLAY statements show values of p, v and z in LP solution
$OFFTEXT
DISPLAY P.L ;
DISPLAY V.L ;
DISPLAY Z.L ;
$ONTEXT
  q(i,t) solution comes from dual - network equation slack values
$OFFTEXT
PARAMETER
      Q(I,T)      evader marginals;
Q(I,T) = 100*(NETL.M(I,T)+NETE.M(I,T)+NETM.M(I,T));
DISPLAY Q ;
$ONTEXT
  u(i,t) computes the maximum score obtainable to the searcher if
    he is in cell i and evader marginals are given
$OFFTEXT
PARAMETER
      ZE(I,T)      longest path by searcher;
ALIAS (T,TP);
ZE(I,T)$(ORD(T) EQ 1)=SUM(TP$(ORD(TP) EQ CARD(T)),Q(I,TP))
LOOP(T$(ORD(T) LT CARD(T)),ZE(I,T+1)=SUM(TP$(ORD(TP)+ORD(T) EQ CARD(T)),
                          Q(I,TP))+MAX(ZE(I-1,T),ZE(I,T),ZE(I+1,T)));
DISPLAY ZE;
```

LIST OF REFERENCES

1. Eagle, J.N., and Washburn, A.R., 1990. Cumulative Search-Evasion Games (CSEGs). *Naval Research Logistics Quarterly* (to appear).

2. Koopman, B.O.,1980. *Search and Screening*. Pergammon Press, New York.

3. Stone, L.D.,1975. *Theory of Optimal Search*. Academic Press, New York.

4. Stone, L.D.,1989. A Review of Results in Optimal Search for Moving Targets. In *Search Theory: Some Recent Developments*, D.V. Chudnovsky and G.V. Chudnovsky (eds.). Marcel Dekker, New York.

5. Eagle, J.N.,1984. The Optimal Search for a Moving Target When the Search Path is Constrained. *Operations Research* **32**, 1107-1115.

6. Stewart, T.J., 1980. Experience with a Branch-and-Bound Algorithm for Constrained Searcher Motion. In *Search Theory and Applications*, K.B. Haley and L.D. Stone (eds.). Plenum Press, New York.

7. Gal, S.,1980. *Search Games*. Academic Press, New York.

8. Ruckle, W.H.,1983. *Geometric Games and their Applications*. Pitman, Boston, p. 156.

9. Stewart, T.J.,1980. A Two-Cell Model of Search for an Evading Target. *European Journal of Operations Research* **8**, 369-378.

10. Robinson, J.,1951. An Iterative Method of Solving a Game. *Annals of Math* **54**, 296-301.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Professor Alan R. Washburn Naval Postgraduate School Code 55Wa Monterey, CA 93943	2
4.	Professor Siriphong Lawphongpanich Naval Postgraduate School Code 55Lp Monterey, CA 93943	1
5.	Professor James N. Eagle Naval Postgraduate School Code 55Er Monterey, CA 93943	1
6.	Dr. Lawrence D. Stone Metron, Inc. 1481 Chain Bridge Road Suite 203 McLean, VA 22101	1
7.	Dr. James Weissinger Daniel H. Wagner, Assoc. Suite 205 894 Ross Drive Sunnyvale, CA 94089	1
8.	Lt. Brian P. Bothwell 64 Colony Drive Westfield, MA 01085	2